Cryptografie

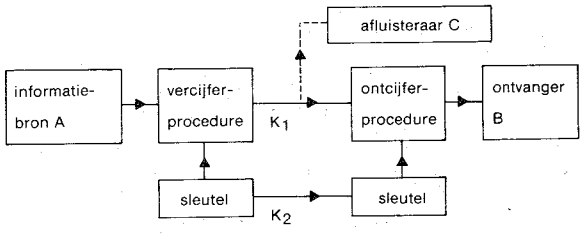
CRYPTOGRAFIE

door prof. dr. J. H. van Lint

In het vorige nummer van Informatie *(juli/augustus 1983) beschreef drs. J. J. Borkink welke strategieën beschikbaar zijn om software te beschermen tegen misbruik of oneigenlijk gebruik. Een van die strategieën is gebaseerd op het gebruik van cryptografische technieken; technieken die bij uitstek geschikt lijken voor computertoepassing, omdat het daarbij immers gaat om 'vercijfering' of 'codering' van boodschappen.*
In dit artikel, een bewerking van een rede uitgesproken ter gelegenheid van de 27ste herdenking van die dies natalis van de Technische Hogeschool Eindhoven, meer over de wiskundige en algoritmische technieken die heden ten dage bij cryptografie gebruikt worden.

1 INLEIDING

De meesten onder u zullen bij het horen van de naam cryptografie vermoedelijk denken aan het klassieke model van communicatie in geheimschrift, beschreven in figuur 1.



Figuur 1

Men denkt hierbij meestal aan communicatie tussen militaire eenheden of tussen diplomatieke diensten e.d. Daarbij is K_1 het communicatie-kanaal dat niet veilig is, d.w.z. het kan worden afgeluisterd door tegenstanders.
De informatie gaat in geheimschrift door dit kanaal. De procedures voor vercijfering en ontcijfering hangen af van een *sleutel* die de zender A eerst via een wèl veilig kanaal K_2 (b.v. via een koerier) naar de ontvanger B moet sturen. Het probleem dat de afluisteraar, die in de theorie een 'cryptanalyst' wordt genoemd, moet oplossen is het breken van de code, d.w.z. dat hij probeert de sleutel te ontdekken. Op dit gebied is zeer veel wiskundig werk gedaan en vooral uit de 2e wereldoorlog zijn enige indrukwekkende prestaties bekend.
Een nogal simpel voorbeeld van een klassiek cryptografisch systeem is de eenvoudige, *mono-alfabetische substitutie* waarbij de letters van het alfabet worden gepermuteerd zoals in

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
T H E K R Z W D L P O B J Q V A M I Y C S X G U N F

Bij deze sleutel wordt DIES omgezet in KLRY. Dit systeem is heel eenvoudig te breken door gebruik te maken van de statistische eigenschappen van de taal. Op verdere details van de klassieke cryptografie gaan we niet in. In het laatste deel van dit artikel wordt het systeem DES genoemd, een voorbeeld van mono-alfabetische substitutie met een alfabet van veel meer dan 26 letters.
Het werk waarover ik wil spreken, is om verschillende redenen nodig geworden:

- telefoongesprekken worden steeds vaker via microgolfstraalzenders en satellieten overgebracht. Het is voor afluisteraars eenvoudig geworden om deze gesprekken ook te horen. Het is bekend dat tenminste één ambassade in Washington het Amerikaanse telefoonverkeer afluistert en dat alle gesprekken, telex en telegrammen, die de Verenigde Staten in- of uitkomen door de National Security Agency (NSA) worden afgeluisterd;
- elektronisch betalingsverkeer tussen banken en elektronisch berichtenverkeer worden steeds algemener. Controle van de herkomst van een opdracht is daarbij essentieel. Er moet een digitaal equivalent van de handtekening onder een cheque zijn. We verstaan daarbij onder een handtekening iets dat slechts door één persoon is te maken, maar door velen kan worden gecontroleerd;
- er komen steeds meer situaties waarin een computer de identiteit van een gebruiker moet controleren, zoals bij de log-in-procedure die toegang geeft tot de machine- en de gegevensbestanden. Men wil voorkomen dat de daarbij gebruikte wachtwoorden uit het geheugen van de computer gestolen worden.

Daar het hoe langer hoe vaker voorkomt dat informatie in digitale vorm wordt overgebracht of opgeslagen is het voor afluisteraars resp. indringers mogelijk de boodschappen door (steeds snellere) computers te laten analyseren.
Bij vele van de nieuwe vormen van communicatie bestaat naast afluisteren het gevaar van de introductie van valse informatie in het kanaal en ook hiertegen moeten we ons wapenen.
Het is niet moeilijk te begrijpen dat cryptografie in de toekomst een belangrijke rol gaat spelen. Er zullen commerciële *'cryptonetwerken'* ontstaan die wellicht duizenden abonnees zullen hebben. Het is niet mogelijk om ieder tweetal gebruikers *vooraf* een aparte geheime sleutel te laten afspreken en het zal vaak voorkomen dat men mededelingen wil sturen naar een gebruiker met wie men niet eerder contact heeft gehad.
Deze inleiding moet genoeg voor u zijn om mijn eigenlijke onderwerp te kunnen waarderen, te weten de in 1976 door W. Diffie en M. E. Hellman geïntroduceerde *'public-key-cryptosystems'*.

2 TRAP-DOOR-ONE-WAY-FUNCTIONS

Het hoofdbestanddeel van de systemen die ik wil uitleggen, zijn de zg. *'trap-door-one-way-functions'* (een hele mond vol). Wat moeten we ons daarbij voorstellen? Laat f een *functie* (een *voorschrift* of wellicht een *programma* voor een rekenmachine) zijn die aan getallen x, uit een zekere collectie X, waarden y = f(x) uit Y toevoegt. We eisen dat het een eenvoudige functie is (b.v. een programma met slechts een paar honderd instructies) en dat f een 1 - 1 afbeelding is (d.w.z. als f(a) = f(b) dan is a = b).

Stel nu dat we bij gegeven y uit Y de rekenmachine willen laten uitzoeken voor welke x uit X geldt dat y = f(x), d.w.z. we willen de inverse functie $f^{\leftarrow}$ bepalen). We eisen dat een programma dat dit realiseert onmogelijk lang is (zeg 10^{10} instructies), zo dat het in het algemeen bij gegeven y een machine honderden jaren rekentijd zou kosten om het origineel x te vinden. In dit geval heet f een 'one-way-function'.

We geven direct een belangrijke toepassing. Bij moderne computersystemen moeten gebruikers niet alleen hun naam, maar ook een geheim 'password' x opgeven, voordat ze echt toegang krijgen tot het systeem. Als nu in de computer een lijst van gebruikers opgeslagen zou zijn met hun respectievelijke passwords, is het goed denkbaar dat deze lijst toch in verkeerde handen terechtkomt (denk aan kwaadwillende onderhoudsmensen, operators, etc.).

Daarom wordt in een modern computersysteem bij elke gebruiker niet zijn password x opgeslagen, maar de waarde van f(x), waarbij f een one-way function is.

Als een gebruiker zijn naam en password x aanbiedt, berekent de machine f(x) en controleert of deze in orde is. Iemand die de lijst van gebruikers met hun f(x) waarde uit het geheugen steelt, heeft daar niets aan, omdat hij, zelfs als hij f weet, de inverse $f^{\leftarrow}$ niet kan bepalen in redelijke tijd.

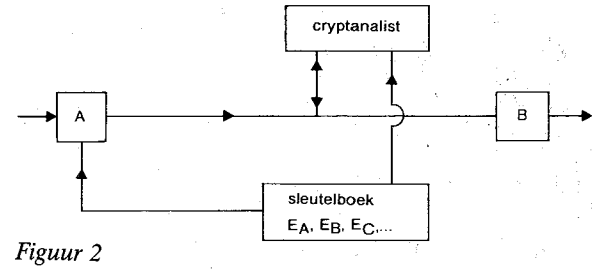
En nu nog de verklaring van de uitdrukking 'trap-door'. Deze naam slaat op een geheim knopje dat nodig is om een anders niet te openen deur open te krijgen. In het geval van onze functie f is het een geringe hoeveelheid extra informatie over f waardoor de berekening van $f^{\leftarrow}$ plotseling eenvoudig wordt. Voorbeelden geef ik verderop.

3 PUBLIC-KEY-CRYPTOSYSTEMS

Nu kan ik de werking van een public-key-cryptosystem uitleggen. Eerst bepaalt iedere gebruiker A zijn eigen paar sleutels E_A, D_A, waarbij E_A (Encryption = vercijfering) een trap-door-one-way-function is en D_A (Decryption = ontcijfering) de bijbehorende ontcijfersleutel (die A kan bepalen omdat hij over de vereiste extra informatie beschikt). Hierbij moet D_A voldoen aan $D_A (E_A (x)) = x$ voor elke boodschap x. Kennis van E_A alleen is weliswaar in principe genoeg om D_A (de geheime ontcijfersleutel) te bepalen maar het kost te veel tijd of het is te duur.

Nu komt de verrassende wending. De te gebruiken voorschriften voor vercijfering van boodschappen worden *openbaar* gemaakt! De lijst is een soort telefoonboek met gebruikers A en hun sleutel E_A. Iedere gebruiker houdt D_A geheim.

Stel nu dat gebruiker A de boodschap x naar B wil sturen. Eerst zoekt hij in de lijst E_B op (dat is dus de vercijfersleutel van de ontvanger!) en dan stuurt hij de boodschap $y = E_B (x)$. Voor B is het eenvoudig om $D_B (y)$ te bepalen daar hij D_B kent. Omdat $D_B (y) = D_B (E_B (x)) = x$ heeft de ontvanger de boodschap ontcijferd. Een afluisteraar die y heeft bemachtigd en weet dat de boodschap voor B is bestemd, kan in de openbare lijst E_B opzoeken, maar daar heeft hij niets aan. Hij kan immers weliswaar *in principe* D_B bepalen, en de boodschappen dan ontcijferen, maar in de praktijk duurt dit veel te lang.



Figuur 2

Als voor alle y geldt dat $E_A (D_A (y)) = y$, (en als E_A een 1-1 afbeelding is dan geldt dit) kan men nu ook boodschappen 'ondertekenen'. Dit gaat als volgt. A stuurt eerst zonder vercijfering zijn naam naar B, zodat B weet dat hij een mededeling van A kan verwachten. Dan zet A via de alleen aan hem bekende functie D_A de boodschap x om in $D_A (x)$. Tenslotte stuurt hij $y = E_B (D_A (x))$. Alleen de ontvanger kan hieruit m.b.v. D_B eerst $(D_A (x)$ bepalen. Dan zoekt hij E_A in de lijst op en bepaalt $E_A (D_A (x)) = x$. Hij is nu absoluut zeker dat de mededeling van A afkomstig is en bovendien kan A later niet ontkennen de boodschap te hebben gestuurd als B zorgvuldig $D_A (x)$ bewaart. Immers, niemand behalve A kan $D_A (x)$ uit x maken. Merk op dat iedereen door $D_A (x)$ in te vullen in $E_A (x)$ kan controleren dat de boodschap door A is verzonden, in overeenstemming met de eerder gegeven definitie van een handtekening.

Bij zeer veel cryptosystemen is het moeilijk om een procedure voor ondertekening te bedenken. In deze richting wordt veel onderzoek gedaan.

3.1 RSA-systeem

Het idee van public-key-cryptography is erg leuk, maar voor we er iets aan hebben, moeten we eerst de daarvoor nodige functies bedenken. In 1978 is door R. L. Rivest, A. Shamir en L. Adleman een systeem bedacht dat nu als het RSA-systeem bekend staat. Enig nadenken moet ik nu verlangen, – maar u hoeft niet te rekenen. We geloven allemaal graag dat een computer dit uitstekend voor ons kan doen. Het volgende schema geeft het RSA-systeem weer. Slechts één voor u wellicht nieuwe notatie moet ik toelichten:

a(mod k) betekent: de *rest* bij deling van a door k,
bv. 33 (mod 7) ≡ 5 omdat 33 = 4 ⋆ 7 + 5.

We stellen ons de te versturen boodschappen voor als getallen x (met zeer veel cijfers).

---

1. Bepaal twee priemgetallen p en q (elk van ± 100 cijfers).
2. Bereken n = pq en m = (p − 1) (q − 1).
3. Kies een getal d (groot) dat onderling ondeelbaar is met m.

4. Bepaal het getal e zó dat e d ≡ 1 (mod m).
Openbare vercijfersleutel: (n, e)
algoritme voor vercijfering: $E_A(x) := x^e \pmod n$.
Geheime ontcijfersleutel: d
algoritme voor ontcijfering: $D_A(y) := y^d \pmod n$.

Figuur 3: RSA-systeem: gebruiker A

De (kleine) stelling van Fermat (1640) luidt:
Als p priem is en a is niet door p deelbaar dan is $a^{p-1} \equiv 1 \pmod p$.
Voorbeeld:
$p = 11$, $a = 2$: $a^{p-1} = 2^{10} = 1024 = 11 \star 93 + 1$.
Gevolg van de stelling is: Als p en q priem zijn en a is onderling ondeelbaar met pq dan geldt
$a^{(p-1)(q-1)} \equiv 1 \pmod{pq}$,
omdat $a^{(p-1)(q-1)} - 1$ zowel door p als door q deelbaar is.
Voor het RSA-systeem vinden we
$D_A(E_A(x)) = (x^e)^d = x^{1+ml} = x(x^m)^l \equiv x \pmod n$.
Iemand die wil afluisteren moet bij gegeven n en e het getal d bepalen.
Daarvoor is volgens regel 4 het getal m nodig. Om m te bepalen moet de afluisteraar eerst p en q vinden, d.w.z. hij moet n in factoren ontbinden.
Aan het probleem van factorisering van getallen is al eeuwen gewerkt.
Vele algoritmen zijn bekend, o.a. diverse redelijk snelle, die in de laatste jaren zijn gevonden dank zij de stimulans van de cryptografie. Voor een getal van 200 cijfers zoals onze n is het aantal bewerkingen nodig om n te ontbinden, volgens de snelste thans bekende methode, groter dan 10^{20} en zelfs de snelste computer is daar na een eeuw rekenen niet mee klaar.
We moeten natuurlijk wel bedenken dat het denkbaar is dat een organisatie waarvoor geheime communicatie belangrijk is, een algoritme heeft gevonden voor veel snellere factorisatie van getallen. In dat geval zullen ze dit zeker niet aan de grote klok hangen. De resultaten van universitair onderzoek komen nog in de literatuur terecht, maar daarover zeg ik dadelijk nog meer.
De publikatie van het RSA-systeem ging gepaard met sensationele krantekoppen zoals: 'The new unbreakable codes – will they put NSA out of business?'
Natuurlijk werd van alle kanten geprobeerd om dit systeem te kraken en wel zonder n eerst te ontbinden. Gedeeltelijk succes van deze pogingen heeft geleid tot verzwaring van de eisen die aan p en q worden opgelegd. Op dit moment ziet het RSA-systeem er nog volkomen veilig uit. Het wordt ook al commercieel gebruikt.
Hoe staat het met werk dat de gebruiker A zelf moet doen? (zie figuur 3).
Het is gewenst dat dit weinig tijd kost, omdat er situaties zijn waarin men de sleutels E_A en D_A regelmatig wil veranderen. De berekeningen in de regels 2, 3 en 4 zijn zeer eenvoudig, maar hoe kom je aan een priemgetal van 100 cijfers? (Korte tijd na het bekend worden van het RSA-systeem was er een Amerikaans bedrijf dat zulke priemgetallen te koop aanbood voor een paar honderd dollar. Erg geheim is je systeem niet als je daar op ingaat!) Als we via een random generator een oneven getal van 100 cijfers kiezen, dan is de kans dat het een priemgetal is ongeveer 1%. Als we snel kunnen controleren of een gekozen getal inderdaad priem is, dan lukt het via proberen dus vrij snel om een paar (p, q) te genereren. Met enige trots kan ik vermelden dat de eerste priem-toets die inderdaad snel werkt (± 45 seconden voor getallen van 100 cijfers), vorig jaar door de Nederlands wiskundige H. W. Lenstra is gerealiseerd op de CDC-computer van het SARA-rekencentrum te Amsterdam. Voor die tijd was er weliswaar een snel algoritme bekend, maar dit gaf nooit 100% zekerheid over al of niet priem zijn.
Een populaire beschrijving van het werk van Lenstra en anderen is in het nummer van Scientific American van december 1982 te vinden.

3.2 **Trap-door-knapsack-system**

Hoewel mij de tijd ontbreekt om op details in te gaan, moet ik voor het vervolg van mijn verhaal toch iets zeggen over het tweede public-key-system. Dit is bekend onder de naam 'trap-door-knapsack-system' en het is bedacht door R. Merkle en M. E. Hellman (1978). Het idee is afkomstig van een bekend probleem uit de combinatoriek. Als een lijst met getallen gegeven is en van een bepaalde deelverzameling wordt de som S gevraagd, dan is dat een eenvoudige optelling. Is daarentegen S gegeven, dan is het in het algemeen zeer lastig om terug te vinden welke getallen zijn opgeteld. De volgende figuur maakt het idee iets duidelijker.

Voorbeeld:	3	5	10	22	43	90	201	
Boodschap	1	0	1	0	0	1	0	
	S = 103 = 90 + 10 + 3 (simpel!)							
Openbare sleutel:	130	49	98	115	19	379	159	
Boodschap	1	0	1	0	0	1	0	
Vercijferde boodschap: 607 = ? + ... + ?								

Figuur 4: Trap-door-knapsack-system

Het voorbeeld boven in de figuur is wel erg flauw. Men ziet bij gegeven S direct hoe de som is gevormd. Bij het daaronder gegeven systeem wordt de boodschap 1010010 omgezet in 130 + 98 + 379 = 607. De afluisteraar die 607 opvangt èn de vercijfersleutel kent, moet even puzzelen om de boodschap te vinden. Dit voorbeeld is nog steeds te eenvoudig, omdat de afluisteraar niet gedwongen wordt alle verschillende mogelijkheden één voor één te proberen. Hij ziet zo dat 607 niet gehaald wordt zonder 379 te gebruiken.
De grap van het knapsack-systeem is dat de openbare sleutel is ontstaan uit het flauwe voorbeeld door een eenvoudige transformatie. De ontvanger voert eerst de omgekeerde transformatie (die hij alleen kent) uit op 607 en vindt dan 103 waarna hij in een wip klaar is. De transformatie in het voorbeeld bestond eruit dat alle getallen in het simpele voorbeeld vermenigvuldigd zijn met 211 en vervolgens gereduceerd mod 503.
In de praktijk is de openbare sleutel een lijst van 100 getallen van ongeveer 30 cijfers.
Het is bekend dat het 'knapsack-problem' zeer moeilijk op te lossen is.
Wat bedoel ik daar mee? Het is duidelijk dat een exacte definitie nodig is.
Chinees spreken lijkt mij heel erg moeilijk, maar aangezien miljoenen kleuters het dagelijks doen, valt het blijkbaar wel mee! Als we het woord 'moeilijk' willen definiëren in de context van de cryptografie, komen we terecht in een ander nog jong gebied van wiskundig onderzoek, nl. de analyse van de hoeveelheid rekentijd en geheugen-

ruimte die nodig is om een bepaald probleem met een computer op te lossen. Dit gebied heet 'computational complexity theory'. Eén van de aspecten van deze theorie is het indelen van problemen in klassen van problemen van vergelijkbare moeilijkheid.
Een beruchte klasse in deze theorie staat bekend als NPC (nondeterministic polynomial complete).
Als voor één probleem uit deze klasse een redelijk snel (d.w.z. de rekentijd is polynomiaal in de parameters van het probleem) algoritme bestaat i.p.v. de nu bekende, langzame (nl. de rekentijd is exponentieel in de parameters van het probleem) algoritmes, dan geldt dit voor de hele klasse!
Het knapsack-probleem behoort tot de klasse NPC en tot nu toe is zo'n snel algoritme niet gevonden.
Laten we echter voorzichtig zijn. In het trap-door-knapsack-systeem is de openbare sleutel helemaal niet willekeurig! Hij is verkregen via een truc (de trap-door) uit een triviale rij (zoals in ons voorbeeld in figuur 4).
Het is nog maar de vraag of het trap-door-knapsack-systeem tot de moeilijke klasse NPC hoort. Deze vraag is beantwoord: Op 12 mei jl. stond op de voorpagina van de Los Angeles Times de kop 'Unbreakable computer code proves otherwise'. De wiskundige A. Shamir had precies gedaan wat al gevreesd was en Merkle en Hellman waren de $ 100 kwijt die zij hadden uitgeloofd voor het breken van een van hun voorbeelden. Korte tijd later stelde de directeur van NSA, de admiraal R. Inman, dat NSA vele jaren voor Diffie en Hellman het idee van public key cryptography had uitgevonden en ook het knapsack-systeem, maar dat ze al snel ontdekt hadden dat dit systeem te breken is.
Hij zei verder dat NSA zich niet verplicht voelt om commerciële instellingen van dit soort kennis op de hoogte te stellen. Insiders beweren dat NSA wel degelijk AT & T heeft gewaarschuwd om de knapsack-methode nooit te gebruiken.

4 MOEILIJKHEDEN

We zijn hiermee aangeland bij een nieuw onderwerp, nl. de vele problemen die zijn ontstaan rond de cryptografie in de laatste jaren. Als eerste voorbeeld noem ik het in 1977 door het National Bureau of Standards (NBS) aangenomen officiële Amerikaanse cryptosysteem: de Data Encryption Standard (DES). Dit systeem bewerkt blokken tekst van 64 bits en zet deze om in andere blokken via een transformatie die afhankelijk is van een sleutel van 56 bits. Het is in feite niet anders dan een mono-alfabetische subsitutie, maar dan met een alfabet van 2^{64} letters. Het hele systeem staat tegenwoordig op één LSI-chip. Er zijn inmiddels vele fabrikanten van elektronische apparatuur die DES inbouwen. Dit klinkt geweldig, maar er is iets vreemds aan de hand. Het voorstel over DES, door NBS in samenwerking met IBM tot stand gekomen is in 1975 bekendgemaakt en het leidde direct tot een stroom van kritiek en een verhitte controverse. Er waren twee belangrijke punten van kritiek. Ten eerste maakt de grootte van de sleutel (56 bits) het systeem kwetsbaar. Na enige te optimistische schattingen gelooft men nu dat een special purpose computer van 50 miljoen dollar gemiddeld twee dagen nodig zou hebben om, door eenvoudig alles te proberen, de sleutel te vinden.
Voor bepaalde afluisteraars is dit bedrag een acceptabele investering!
Alle partijen zijn het er inmiddels over eens dat het systeem DES binnen 10 jaar volkomen onveilig zal zijn. Het merkwaardige is dat vele wiskundigen direct hebben aangedrongen op een sleutel van 128 bits en die sleutel zou over 100 jaar nog steeds afdoende zijn. Het wordt nog opmerkelijker als men verneemt dat de NSA weigert om exportvergunning te verlenen aan cryptosystemen waarbij een sleutel van meer dan 64 bits kan worden gebruikt.
Het tweede punt van kritiek betrof het apparaat zelf. De substitutie wordt geregeld door zg. 'S-dozen', waarvan om duistere redenen geheim wordt gehouden hoe die zijn gekozen. Bij een onderzoek van DES is ontdekt dat er een zekere structuur in de S-dozen zit, hoewel officieel is gezegd dat ze willekeurig waren gekozen. Velen vrezen dat ook hier een 'trap-door' is ingebouwd die het voor NSA mogelijk maakt alles te ontcijferen wat elders als veilig opgeborgen wordt beschouwd. Ik geloof het graag.
Het is snel duidelijk geworden dat de NSA het helemaal niet leuk vindt dat aan de universiteiten plotseling met grote interesse aan cryptografie wordt gewerkt. Kort na de introductie van public-key-cryptografie ontstond een rel toen een medewerker van NSA op eigen initiatief een aantal deelnemers aan een congres over cryptografie schriftelijk erop wees dat publikatie van hun resultaten in strijd zou zijn met de Internnational Traffic and Arms Regulation. Daarin wordt gesteld dat uitvoer van 'cryptographic equipment' onder exportcontrole valt. Deze bedreiging van de academische vrijheid was voor velen niet acceptabel. De zaak is eerst door NSA gesust, hoewel deze organisatie wel stelde dat het werk van sommige onderzoekers een bedreiging was van de veiligheid van de Verenigde Staten. Toen is door de American Council on Education een studiegroep gevormd om dit probleem te bespreken. Deze groep heeft een aantal voorstellen van admiraal Inman verworpen, maar heeft tenslotte wel aanbevolen dat auteurs van artikelen over cryptografie hun werk eerst door NSA laten beoordelen. Het gebeurt inmiddels regelmatig, maar niet algemeen. Er zijn al twee artikelen niet gepubliceerd op verzoek van NSA. Onlangs vertelde Martin Hellman mij dat hij er nu zelf ook veel genuanceerder over denkt. Zo had hij zich tijdens de crisis met de gijzelaars in Iran gerealiseerd dat het misschien niet zo verstandig is om iedereen op de wereld te tonen hoe ze hun communicatiesystemen kunnen beveiligen. Bovendien zijn allerlei andere takken van wetenschappelijk onderzoek al lang aan regelingen onderworpen i.v.m. de staatsveiligheid. Wiskundigen zullen moeite hebben hieraan te wennen. Het is in ieder geval een probleem dat discussie waard is.
Naast cryptografie zijn er vele andere gebieden van wiskundig onderzoek sinds kort tot bloei gekomen, vaak onder invloed van allerlei technologische ontwikkelingen. Er is nog veel dat we niet doorzien en er rest een enorme hoeveelheid fascinerend onderzoek. De vrees bestaat dat de onderzoekers aan onze universiteiten niet meer de tijd zullen krijgen om aan deze uiterst belangrijke problemen te werken. Daar zal ons land als het te laat is, heel veel spijt van krijgen.

5 LITERATUUR

1. Diffie W. and M. E. Hellman, New directions in cryptography, IEEE Trans. Information Theory, IT-22 (1976), 644-654.

2. Handelman G. H., Cryptographic research and the national security, SIAM News *14*[3] (1981).
3. Lempel A., Cryptology in transition: a survey, ACM Computing Surveys *11* (1979), 285-303.
4. Merkle R., Secure communication over insecure channels, Communications ACM *21* (1978), 294-299.
5. Merkle R., and M. E. Hellman, Hiding information and signatures in trap-door knapsacks IEEE Trans. Information Theory IT-*24* (1978), 525-530.
6. Pomerance, C., The search for prime numbers, Scientific American, Vol. 247, nr. 6, (December 1982), 122-131.
7. Rivest R. L., A Shamir and L. Adleman, On digital signatures and public key cryptosystems, Communications ACM *21* (1978), 120-126.
8. Hardy and Wright, Theory of numbers, Oxford 1945, Theorem 7.1

BOEK VAN DE MAAND: KIJK AUTOMATISERING

In de boekwinkel ligt gedurende de periode 1 september-1 oktober 1983 het boek 'KIJK AUTOMATISERING, Over mensen, computer en vooruitgang' voor f 14,90 te koop. (na 1 oktober f 26,50).
Het boek is geschreven door prof. J. M. van Oorschot, die naast een aantal andere functies ook lid is van het bestuur van het Nederlands Genootschap voor Informatica.
Het boek behandelt op een voor ieder begrijpelijke wijze de ontwikkeling van het automatiseringsproces door de eeuwen heen. Ook de invloed van de automatisering op onze toekomstige maatschappij komt aan de orde. De vele illustraties en de daarbij behorende teksten geven veel informatie en verlevendigen het boek.
Voor de meeste lezers van INFORMATIE brengt het boek weinig nieuws. Maar het is wel een boek dat aan te bevelen is om kado te geven aan hen die trachten uw vak te begrijpen maar u slaagde er niet in om dat ook daadwerkelijk te doen. Aan de uitgave van het boek is tevens een prijsvraag gekoppeld; waarmee een IBM personal computer te winnen is.